Axioms for bigraphical structure

Math. Struct. in Comp. Science (2005), *vol.* 15, *pp.* 1005–1032.
doi:10.1017/S0960129505004809 Printed in the United Kingdom

Axioms for bigraphical structure

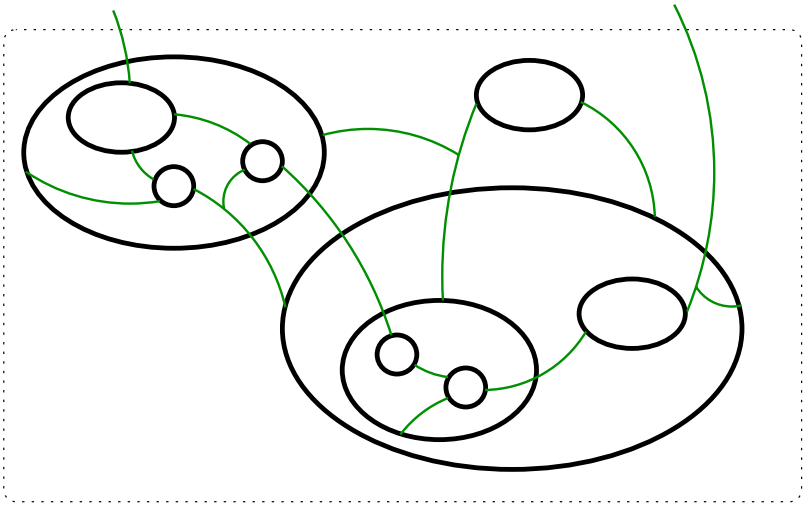
ROBIN MILNER

University of Cambridge, The Computer Laboratory, J J Thomson Avenue, Cambridge CB3 0FD, UK

Received 18 February 2004; revised 9 January 2005

This paper axiomatises the structure of bigraphs, and proves that the resulting theory is complete. Bigraphs are graphs with double structure, representing locality and connectivity. They have been shown to represent dynamic theories for the π-calculus, mobile ambients and Petri nets in a way that is faithful to each of those models of discrete behaviour. While the main purpose of bigraphs is to understand mobile systems, a prerequisite for this understanding is a well-behaved theory of the structure of states in such systems. The algebra of bigraph structure is surprisingly simple, as this paper demonstrates; this is because bigraphs treat locality and connectivity orthogonally.

1. Introduction



The diagram shows a bigraph, suppressing some of its detail. The ovals and circles are *nodes*, which may be nested, and each node has *ports*, which may be linked. The links are a partition of all the ports in a bigraph. Some links are external, shown here with wires escaping from the top of the diagram. When this bigraph is inserted in another (insertion will be represented by categorical composition) it will be placed in some region of the host graph, and each external link joined to some link of the host in a way that does not depend on the placing. Thus the independence of the linking and the placing of nodes, already illustrated by the way links cross boundaries in the diagram, is also respected by the operation of composing bigraphs.

The bigraphical model, which has recently stabilised (Jensen and Milner 2004), arises from a long-term effort (Milner 2001b; Milner 2001a), beginning with action calculi (Milner 1996; Gardner 2000), to provide a theory common to different process calculi, and to base this theory on the topographical ideas that appear to pervade those calculi. These ideas are especially evident in the calculus of mobile ambients (Cardelli and Gordon 2000); less obviously, they have also been found to inform the π-calculus (Milner *et al.* 1992). A contributory effort (Sewell 1998; Cattani *et al.* 2000; Leifer and Milner 2000; Leifer 2001; Milner 2001a; Jensen and Milner 2004) has been to unify the treatment of labelled transition systems by treating the labels as contexts, especially graphical contexts. This unification has recovered existing behavioural theories for the π-calculus (Jensen and Milner 2003) and for mobile ambients (Jensen 2004), and has contributed to that for Petri nets (Milner 2004).

In the categorical treatment of transition labels as contexts, the definition of a transition $a \xrightarrow{L} a'$ of an agent a specifies, among other things, that the composition $L \circ a$ (a in context L) performs a reaction; in other words, a and L may collaborate to perform it. This immediately suggests a categorical formulation in which the arrows are bigraphs, and the objects are interfaces explaining what kind of 'hole' in L will be occupied by the agent a, and what links will connect a to L. Considering the static structure alone, this recalls Lawvere's categorical treatment of algebraic theories (Lawvere 1963), in which the objects are simply finite ordinals and the arrows are tuples of terms; composition is substitution of terms for variables in other terms. In bigraphs, term substitution is replaced by a more ramified notion of graph substitution.

The topic of this paper is to axiomatise the resulting structure of bigraphs. The justification for such a specific topic is threefold. First, the work already cited gives ample evidence that a graphical structure combining topography with connectivity has wide application in computer science; for, as already mentioned, it brings unity to at least three models of discrete dynamics, each of which already has many applications. Second, the algebraic treatment of such dual structures does not appear to have been addressed previously; yet the behaviour of systems whose connectivity and topography are both reconfigurable may be so complex that their dynamics cannot be properly understood without a complete and rigorous treatment of their statics. Bigraphs are just one possible treatment of such dual structure, but it is likely that their static theory can be modified for other treatments. Third, as we shall see, dual structures seem to require a novel kind of normal form, which is essential to a proof of axiomatic completeness.

Underlying this dual structure is the notion of *strict symmetric monoidal category*, which is growing in importance in the foundation of computation, especially as concurrency becomes more prominent. An early use of strict monoidal categories was by Benson (Benson 1975), who drew attention to them as the natural foundation for syntactic derivations. He even proposed that the free strict monoidal category warrants the title 'the syntactic category'. This insight survives the change of emphasis from *derivations*, where we think of generating a language from a grammar, to *rewritings*, which are used to represent the dynamic behaviour of a system. Evidence for this is in the paper 'Petri nets are monoids' (Meseguer and Montanari 1990). Bigraphs represent a further step in this direction; they can be considered as the *syntax*, but better as the *static structure*, found

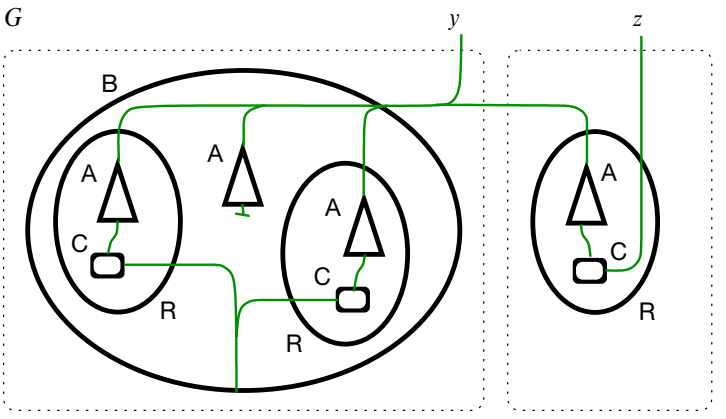


Fig. 1. A bigraph G showing agents interacting in a structured environment.

in systems whose chief purpose is to manipulate their own structure. To accommodate common features of such systems, they have become a strict *symmetric* monoidal category, which is nearly, but not quite, free. This paper characterises that structure.

We begin in Section 2 with examples, whose purpose is to justify the structure in terms of the kind of systems it can model. One example illustrates a simple mobile real-life system, and others are drawn from process calculi. In Section 3 we set up bigraphs as a category, and in Section 4 we explore their algebra far enough to be able to propose the notion of *discrete normal form* (DNF). Section 5 introduces axioms, which are based upon those of a strict symmetric monoidal category, and proves their completeness using DNF. Section 6 introduces an alternative, *connected normal form* (CNF), that appears closer to process calculi and to programming languages, but uses a form of product that lends itself less well to axiomatisation. Finally, the concluding section mentions some related work and open problems.

2. Examples

This section reviews how bigraphs may be applied. We begin by illustrating their structure, and how they may be composed, reflecting as far as possible how real systems are built from subsystems. We then give four examples of reaction rules, the means by which bigraphical systems reconfigure themselves. The main intention of the bigraph model is to support dynamic behaviour as apparently different as physical systems (such as our structural example) and process calculi, and thereby to reveal and study what they have in common. This dynamic study is treated in detail elsewhere, and will be the subject of further publications; the examples in this section are given to motivate their underlying static structure, which is the main topic of the present paper.

The bigraph G in Figure 1 is a snapshot of part of a system in which people and things are interrelated and interacting. A bigraph consists principally of *nodes*, shown here with bold outlines. (Their differing shapes are meant to be informally helpful, but are formally insignificant.) The nodes may be nested, and they have *ports* which may be connected

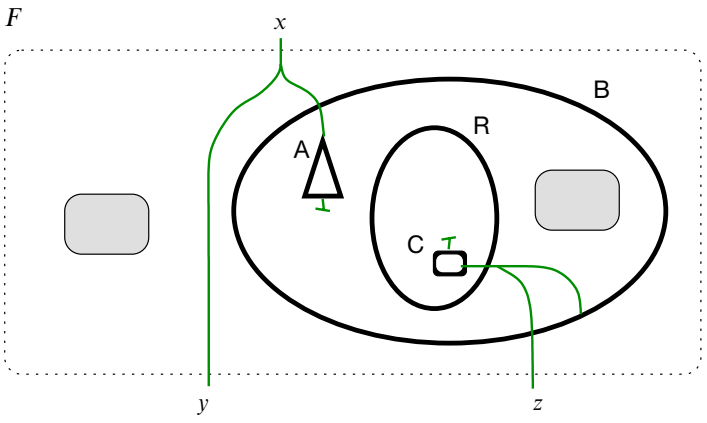


Fig. 2. A host context F.

by *links*. Each link may connect many ports; for example, all the A-nodes are joined by a single link represented by a forking bent line. The term 'bigraph' connotes this double structure of nesting and linking. The nesting of nodes imposes no constraint upon the linkage of their ports.

Every node is assigned a *control* (A, B, ...) which represents what kind of node it is and determines its *arity*, that is, how many ports it has. In our example, the arities of A, B, C and R are 2, 1, 2 and 0, respectively. Here A-nodes represent agents, perhaps people equipped with mobile phones; the link among them indicates that they are conducting a conference call. If an agent is in a room (R) it may also be plugged in to a computer C. The computers in all rooms of a building (B) may be connected by a network, part of the building's infrastructure.

Our bigraph represents only part of a total (host) system. Some links have names; these are the *open* links that may be connected to other parts of the host. On the other hand the *closed* links, such as those joining three agents to their computers or two computers to the building, cannot be connected to other parts.

Every bigraph has a *width*; this one has width 2, shown by the two dotted rectangles, called *regions*. Nodes in the same region cannot be located within different nodes of the host; on the other hand, different regions may be arbitrarily far apart in the host.

As in all computing structures, some aspects of the behaviour of a subsystem can be analysed independently of its position within, or linkage to, its host system. In this case, we can imagine analysing the exchange of information among the four agents, and how it relates to their interaction with their computers, even though we know neither what building is occupied by the fourth agent, nor what other agents in the host are also taking part in the conference call (via the open link y).

Figure 2 illustrates a host bigraph F, which G may inhabit. F is a *contextual* bigraph, in two ways. It has two *holes*, the grey squares, which may be inhabited by a bigraph with two regions (such as G has). It also has two *inner names*, y and z, for linking to an inhabitant bigraph via its corresponding *outer names*. We draw a bigraph's inner names

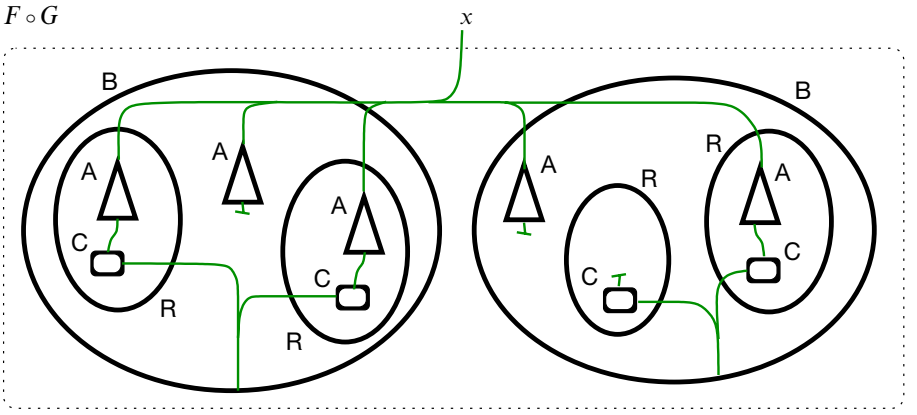


Fig. 3. $F \circ G$, the partial system G embedded in host context F.

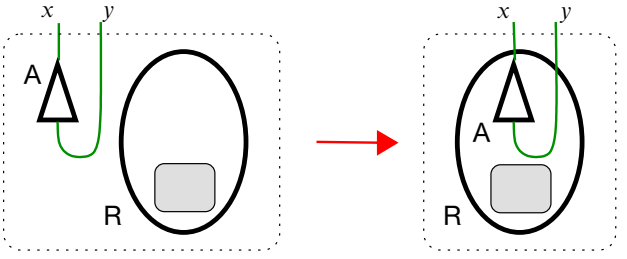


Fig. 4. An agent enters a room.

below it, and its outer names above it. An inner name cannot be associated with a particular hole, because the corresponding link in an inhabitant belongs to no particular one of its regions; a link has no location (though each of its ports does).

Figure 3 shows the result $F \circ G$ of inserting G into the context F. We now know more about the conference call; it is being conducted by three people in one of the two buildings and two in the other. But $F \circ G$ may inhabit a further context; so, via its open link x the conference call may involve inhabitants of further buildings. It may not be obvious at first that the bigraph in Figure 3 represents the insertion of G into F (to see it, *first* place each region of G in the proper hole of F and *then* join corresponding inner and outer names); the reason is that placing and linking are in a sense orthogonal. This will be reflected in our mathematical formulation.

We turn now to reconfigurations, or *reactions*. Different controls are equipped with different *reaction rules*, often called *rewriting rules* in syntactic or graphical systems. Each rule consists of a precondition or *redex*, which may be transformed into a postcondition or *reactum* wherever it occurs; both of these are themselves bigraphs. The first two reaction rules pertain to the preceding example.

Figure 4 allows an agent A, in the same region as a room R, to enter the room. The hole, or *parameter*, in the redex represents all other possible occupants (computers, agents, . . .)

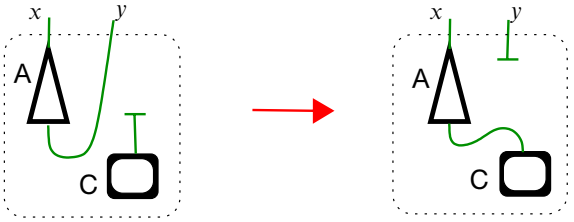


Fig. 5. A computer connects to an agent.

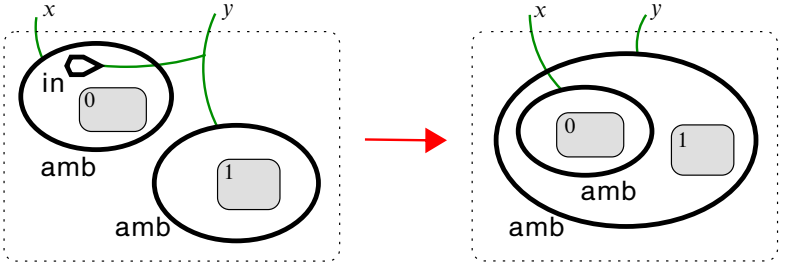


Fig. 6. One mobile ambient enters another.

of the room. This rule is purely a placing reconfiguration. The linking persists; it represents any possible connectivity (possibly none) between the agent and the host context.

Figure 5 allows an agent A in the same region (perhaps a room) as a computer C, to become linked to the computer. This rule is purely a linking reconfiguration. Note that the rule requires the computer to be unlinked beforehand; on the other hand, if the agent is linked to a computer (elsewhere?) beforehand, he or she exchanges any such link for the new one. A variant of this rule could require the agent to be unlinked beforehand.

Figure 6 is a slight simplification of a rule for the calculus of mobile ambients (Cardelli and Gordon 2000). Note that each ambient, unlike the room in Figure 4, is named (by a link); this allows the command 'in', residing within ambient x but referring to ambient y, to cause ambient x and all its contents to enter ambient y. Ordinals 0,1 identify parameters in the reactum with those in the redex.

Finally, Figure 7 represents the single rule of communication in the π-calculus (Milner *et al.* 1992), simplified for pure CCS, where it would be written

$$(\overline{x}.P + M) \,|\, (x.Q + N) \longrightarrow P \,|\, Q.$$

The parameters M and N represent alternative actions of the same form as $\overline{x}.P$ or $x.Q$. The ordinals 0,1,2 and 3 represent P, M, Q and N; thus, as in CCS, the effect of the communication is to discard the alternatives in each sum. (The standard rule for the π-calculus is slightly more refined.) An important role of placing emerges here. The controls B, R and amb in the previous examples are all declared as *active*, which means that other reactions can occur inside them; on the other hand, the controls sum, send and get are all *passive*, thus preventing internal reaction. This matches the intention in CCS that

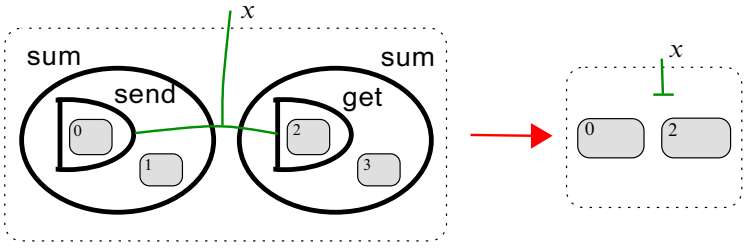


Fig. 7. A communication occurs in CCS.

summation '+' and prefixing '$\overline{x}.$' and '$x.$' are *guarding*; their reactions must precede those of their contents.

The dynamical theory of bigraphs has been somewhat developed (Jensen and Milner 2004), and shown to model behaviour in the π-calculus (Jensen and Milner 2003), mobile ambients (Jensen 2004) and Petri nets (Milner 2004), in each case making a theoretical contribution to those disciplines. This development rests firmly on the static structure that is the main topic of the present paper.

3. Definitions

In this section we define the notion of *bigraph* formally in terms of the constituent notions of *place graph* and *link graph*. We shall organise bigraphs as the arrows of a partial strict symmetric monoidal category whose objects are a simple form of interface. (We explain what 'partial' means in a footnote below.)

Before we look at the details, it is helpful to have a simple insight into how to build a bigraph. Take a collection of *nodes*, each having several *ports*, each of which has a wire hanging loose. We build a bigraph in two stages:

(1) *Placing*: nest nodes inside each other in any way you please, with no concern for the wires.
(2) *Linking*: join wires together in any way you please, with no concern for the nesting.

All that remains are the subtleties of how to make a larger bigraph from smaller ones.

Notation. We use '$\circ$', 'id' and '$\otimes$' for composition, identity and tensor product in a category. Id_S denotes the identity function on a set S, and $\varnothing_S$ the empty function from $\varnothing$ to S. We use $S \uplus T$ for the union of sets S and T that are known or assumed to be disjoint, and $f \uplus g$ for the union of functions with domains that are known or assumed to be disjoint. We often interpret a natural number m as a finite ordinal $m = \{0, 1, \ldots, m-1\}$. We use $\vec{x}$ to denote a finite sequence $\{x_i \mid i \in m\}$. We presuppose a denumerable set $\mathcal{X}$ of *names*.

Definition 3.1 (Signature). A *signature* $\mathcal{K}$ is a set whose elements are called *controls*. For each control $\mathsf{K} \in \mathcal{K}$ it provides a finite ordinal $ar(\mathsf{K})$, which is an *arity*.

In refinements of the theory, a signature may carry further information, such as a *sign* and/or a *type* for each port. The sign may be used, for example, to enforce the

restriction that each negative port is connected to exactly one positive port, as in action calculi (Cattani *et al.* 2000; Milner 1996). Another possible refinement is a *kind* assigned to each node, determining the controls of the nodes it may contain. (An extreme case would be *atomic* nodes, which may contain no other nodes at all.)

Definition 3.2 (Interface). An *interface* $I = \langle m, X \rangle$ consists of a finite ordinal m called a *width* and a finite set $X \subset \mathcal{X}$ called a *name set*[†].

We are now ready to define bigraphs. We shall first define a *concrete* bigraph to be the combination of two constituents, a place graph and a link graph.

Definition 3.3 (Concrete bigraph). A *concrete bigraph* over the signature $\mathcal{K}$ takes the form $G = (V, E, ctrl, G^{\mathsf{P}}, G^{\mathsf{L}}) : I \to J$, where the interfaces $I = \langle m, X \rangle$ and $J = \langle n, Y \rangle$ are its *inner* and *outer faces*. Its first two components V and E are finite sets of *nodes* and *edges*, respectively. The third component, $ctrl : V \to \mathcal{K}$, is a *control map*, which assigns a control to each node. The remaining two components are:

$$G^{\mathsf{P}} = (V, ctrl, prnt) : m \to n \quad \text{a } \textit{place graph}$$
$$G^{\mathsf{L}} = (V, E, ctrl, link) : X \to Y \quad \text{a } \textit{link graph}.$$

We shall define the composition, identities and tensor product of bigraphs in terms of the same operations on their constituents. Let us take place graphs first.

Definition 3.4 (Place graph). A *place graph* $G = (V, ctrl, prnt) : m \to n$ has an *inner width* m and an *outer width* n, both finite ordinals; a finite set V of nodes with a control map $ctrl : V \to \mathcal{K}$; and a *parent map* $prnt : m \uplus V \to V \uplus n$. The parent map is *acyclic*, that is, $prnt^k(v) \neq v$ for all $k > 0$ and $v \in V$.

Being acyclic, the parent map *prnt* represents a forest of n unordered trees. The widths m and n of $G : m \to n$ index its *sites* $0, \ldots, m-1$ and *roots* $0, \ldots, n-1$, respectively. The sites and nodes, that is, the domain of *prnt*, are called *places*. A node or root is *barren* if it has no children.

The sites and roots provide the means for composing the forests of two place graphs; each root i of the first is planted in site i of the second. Figure 8 shows a simple example of composing place graphs; note the correspondence between the sites of F and the roots of G. Formally, let $G_i = (V_i, ctrl_i, prnt_i) : m_i \to m_{i+1}$ $(i = 0, 1)$ be place graphs with $V_0 \cap V_1 = \varnothing$. Then $G_1 \circ G_0 \stackrel{\text{def}}{=} (V, ctrl, prnt)$, where $V = V_0 \uplus V_1$, $ctrl = ctrl_0 \uplus ctrl_1$ and

$$prnt = (\mathsf{id}_{V_0} \uplus prnt_1) \circ (prnt_0 \uplus \mathsf{id}_{V_1}).$$

The identity place graph at m is $\mathsf{id}_m \stackrel{\text{def}}{=} (\varnothing, \varnothing_{\mathcal{K}}, \mathsf{id}_m) : m \to m$.

[†] We saw in the previous section that alphabetic names are used to link bigraphs together. A more abstract presentation would dispense with them, representing names positionally – as in abstract treatments of the λ-calculus. This is perfectly possible here. Indeed, the word 'partial' used above would then be redundant; it refers to the fact that the tensor product of interfaces is defined only when their name sets are disjoint, and that condition is unnecessary with positional notation. We prefer alphabetic names here; it makes little difference to the mathematics, and allows a much more lucid connection to be made to process calculi.

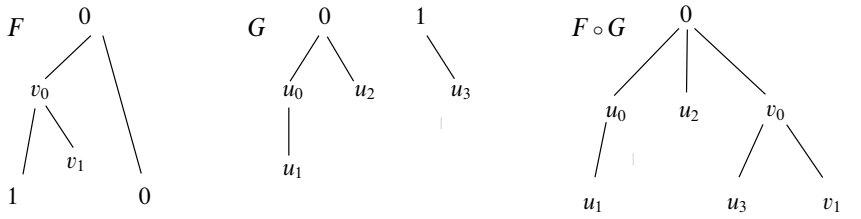


Fig. 8. Composing two place graphs.

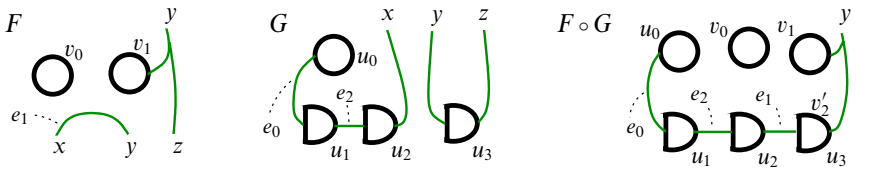


Fig. 9. Composing two link graphs.

The tensor product of two place graphs $F : k \to \ell$ and $G : m \to n$ with disjoint node sets is $F \otimes G : k+m \to \ell+n$. It consists simply of placing the two forests side-by-side, and we need not define it more formally.

We turn now to link graphs.

Definition 3.5 (Link graph). A *link graph* $G = (V, E, ctrl, link) : X \to Y$ has finite sets X of *inner names*, Y of *outer names*, V of *nodes* and E of *edges*. It also has a function $ctrl : V \to \mathscr{K}$, called the *control map*, and a function $link : X \uplus P \to E \uplus Y$, called the *link map*, where the disjoint sum $P \stackrel{\text{def}}{=} \sum_{v \in V} ar(ctrl(v))$ is the set of *ports* of G.

The inner names X and ports P are the *points* of G, and the edges E and outer names Y are its *links*. The link map induces a partition of the points; thus a link graph is a species of hypergraph. A link is *idle* if it has no preimage under the link map. A link graph is *lean* if it has no idle edges. An outer name is an *open* link; an edge is a *closed* link. A point (that is, an inner name or port) is *open* if its link is open; otherwise it is *closed*.

It may seem superfluous to admit the possibility of an idle link, including one that is open, which is an outer name y that 'names' nothing. But the dynamic rules of a bigraphical system allow such a name to arise. In the π-calculus and CCS (see Figure 7) it arises when a communication on the channel y has occurred and no further use of this channel remains. A similar reaction can give rise to a barren root in a place graph.

Figure 9 shows a simple example of composing link graphs. Formally, let

$$G_i = (V_i, E_i, ctrl_i, link_i) : X_i \to X_{i+1} \quad (i = 0, 1)$$

be two link graphs, with $V_0 \cap V_1 = E_0 \cap E_1 = \varnothing$. Then $G_1 \circ G_0 \stackrel{\text{def}}{=} (V, E, ctrl, link)$, where $V = V_0 \uplus V_1$, $ctrl = ctrl_0 \uplus ctrl_1$, $E = E_0 \uplus E_1$ and

$$link = (\mathsf{Id}_{E_0} \uplus link_1) \circ (link_0 \uplus \mathsf{Id}_{P_1}).$$

We can describe the composite link map *link* of $G_1 \circ G_0$ as follows, considering all possible arguments $p \in X_0 \uplus P_0 \uplus P_1$:

$$link(p) = \begin{cases} link_0(p) & \text{if } p \in X_0 \uplus P_0 \text{ and } link_0(p) \in E_0 \\ link_1(x) & \text{if } p \in X_0 \uplus P_0 \text{ and } link_0(p) = x \in X_1 \\ link_1(p) & \text{if } p \in P_1. \end{cases}$$

The identity link graph at X is $\mathsf{id}_X \stackrel{\text{def}}{=} (\varnothing, \varnothing, \varnothing_{\mathscr{K}}, \mathsf{Id}_X) : X \to X$.

The tensor product of two link graphs $F: W \to X$ and $G: Y \to Z$ can be formed provided that their node sets and edge sets are disjoint and that $W \cap Y = X \cap Z = \varnothing$. It is $F \otimes G: W \uplus Y \to X \uplus Z$, and consists simply of the union of their link maps.

We are now ready to define the category that is the main object of study in this paper. Recall that in Definition 3.3 a concrete bigraph $G: \langle m, X \rangle \to \langle n, Y \rangle$ consists of a combination of a place graph $G^{\mathsf{P}}: m \to n$ and a link graph $G^{\mathsf{L}}: X \to Y$ having the same node set and control map. We shall write such a combination as $G = \langle G^{\mathsf{P}}, G^{\mathsf{L}} \rangle$.

Definition 3.6 (Monoidal category of bigraphs). The composition of two concrete bigraphs $G = \langle G^{\mathsf{P}}, G^{\mathsf{L}} \rangle : I \to J$ and $F = \langle F^{\mathsf{P}}, F^{\mathsf{L}} \rangle : J \to K$ with disjoint node sets and disjoint edge sets is

$$F \circ G \stackrel{\text{def}}{=} \langle F^{\mathsf{P}} \circ G^{\mathsf{P}}, F^{\mathsf{L}} \circ G^{\mathsf{L}} \rangle : I \to K.$$

Two concrete bigraphs G_0 and G_1 are said to be *lean-support equivalent*, $G_0 \eqslantless G_1$, if they differ only by a bijection between their nodes and between their non-idle edges; idle edges are ignored. An *abstract bigraph* consists of a $\eqslantless$-equivalence class of concrete bigraphs. Composition and identity of abstract bigraphs are given by

$$\begin{aligned} [F]_{\eqslantless} \circ [G]_{\eqslantless} &\stackrel{\text{def}}{=} [\langle F^{\mathsf{P}} \circ G^{\mathsf{P}}, F^{\mathsf{L}} \circ G^{\mathsf{L}} \rangle]_{\eqslantless} \\ \mathsf{id}_{\langle m, X \rangle} &\stackrel{\text{def}}{=} [\langle \mathsf{id}_m, \mathsf{id}_X \rangle]_{\eqslantless}. \end{aligned}$$

The tensor product of two interfaces with disjoint name sets is

$$\langle m, X \rangle \otimes \langle n, Y \rangle \stackrel{\text{def}}{=} \langle m+n, X \uplus Y \rangle.$$

The tensor product of two abstract bigraphs $F: H \to I$ and $G: J \to K$, where $H \otimes J$ and $I \otimes K$ are defined, is given by

$$[F]_{\eqslantless} \otimes [G]_{\eqslantless} \stackrel{\text{def}}{=} [\langle F^{\mathsf{P}} \otimes G^{\mathsf{P}}, F^{\mathsf{L}} \otimes G^{\mathsf{L}} \rangle]_{\eqslantless} : H \otimes J \to I \otimes K.$$

Figure 10 shows the composition of two bigraphs; they are the combinations of the place graphs and link graphs composed in Figures 8 and 9. The labelling of sites in F indicates where each root of a client bigraph (such as G) should be planted. For clarity, nodes are identified in the figure; in an abstract bigraph these identifiers are forgotten.

The reader may wonder whether the algebra of bigraphs can be factored into two separate algebras, one for placing and the other for linking. But in each of these separate algebras the identity of nodes would have been forgotten (just as in ordinary algebra the identity of subterms is forgotten when one term is substituted in another), and we have seen that the combination of a place graph with a link graph to form a bigraph depends crucially on the identity of nodes. Much of the subtlety of bigraph algebra stems from

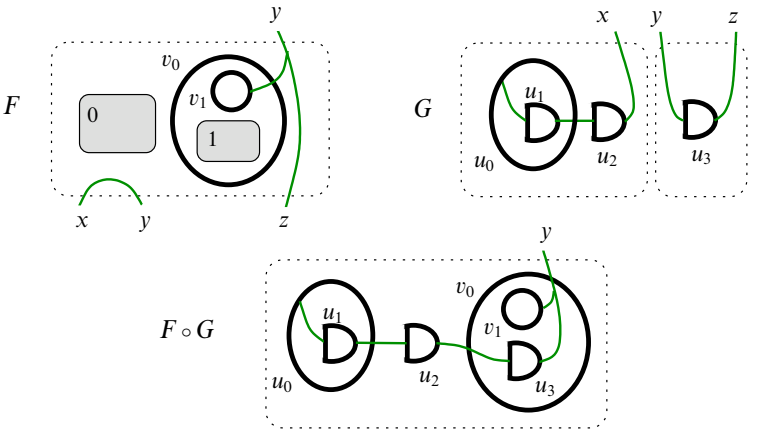


Fig. 10. Composing two bigraphs.

how this combination interacts with the algebraic operations of composition and tensor product.

4. Algebra and discrete normal form

For the remainder of this paper we are concerned only with abstract bigraphs, that is, with $\eqdot$-equivalence classes of concrete ones. We often talk of a class *generated* by certain elements; this means the class formed from those elements using composition and product.

In this section we give the elementary bigraphs from which all others can be generated. We then define a normal form for bigraphical expressions, called *discrete normal form* (DNF), and show how each bigraph can be expressed uniquely in DNF (up to isomorphism).

To avoid too many parentheses in expressions, we shall often represent composition by juxtaposition; it binds tightly, for example $G_1G_2 \otimes G_3$ means $(G_1 \circ G_2) \otimes G_3$.

There are two degenerate forms of interface: a *place interface* $\langle m, \varnothing \rangle$, and a *link interface* $\langle 0, X \rangle$. We write them as m and X (or just x if $X = \{x\}$), respectively. The fully degenerate form is the *origin* $\epsilon \stackrel{\text{def}}{=} \langle 0, \varnothing \rangle$, which is of course the unit for tensor product on interfaces. An important class of bigraphs are the *ground bigraphs* $G : \epsilon \to I$, those whose inner face is the origin. If G is ground, it has no holes or inner names; in this case there is no useful composition GF, since (by the properties of a strict monoidal category) it can be written as a product $G \otimes F$.

A *placing* is a bigraph $m \to n$ with no nodes. All placings can be expressed in terms of three kinds:

$$\begin{array}{rcll} 1 & : & \epsilon \to 1 & \text{a barren root} \\ \mathit{join} & : & 2 \to 1 & \text{join two sites} \\ \gamma_{m,n} & : & m+n \to n+m & \text{swap } m \text{ with } n \text{ places.} \end{array}$$

We use π, ρ to range over *permutations*, those placings generated from the $\gamma_{m,n}$. By contrast, *join* generates placings that join any number of sites.

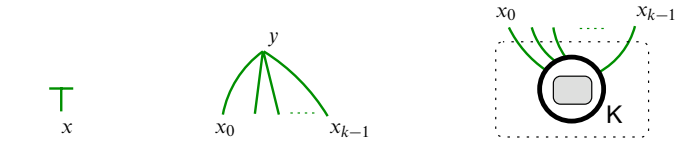


Fig. 11. Elementary linkings and ions.

Definition 4.1 (Merge). For all $m \geqslant 0$ we define $merge_m : m \to 1$ recursively, by

$$\begin{aligned} merge_0 &\stackrel{\text{def}}{=} 1 \\ merge_{m+1} &\stackrel{\text{def}}{=} join(\mathsf{id}_1 \otimes merge_m). \end{aligned}$$

Note that $merge_1 = \mathsf{id}_1$, and hence $merge_2 = join$.

A *linking* or *wiring* is a bigraph $X \to Y$, which necessarily has no nodes. All linkings can be expressed in terms of two kinds (see Figure 11):

$$\begin{aligned} /x &: x \to \epsilon \quad \text{closure} \\ {}^{y}/X &: X \to y \quad \text{substitution } x \mapsto y \text{ (all } x \in X). \end{aligned}$$

A closure just closes a single link. For $X = \{x_1, \ldots, x_k\}$ we define the multiple closure $/X \stackrel{\text{def}}{=} /x_1 \otimes \cdots \otimes /x_k$. For $X = X_1 \uplus \cdots \uplus X_n$ and $Y = \{y_1, \ldots, y_n\}$, a multiple substitution $\sigma : X \to Y$ is defined by ${}^{y_1}/X_1 \otimes \cdots \otimes {}^{y_n}/X_n$. A substitution need not be surjective. We write $Y : \epsilon \to Y$ for the empty substitution (that is, when X is empty), or just $y : \epsilon \to y$ if $Y = \{y\}$; these are the duals of closures. We shall use ω to range over linkings, σ, τ over substitutions, and α, β over the bijective substitutions, which we call *renamings*.

Permutations π and renamings α, together with the identities, generate all isomorphisms in the category of bigraphs; in fact every isomorphism takes the form $\pi \otimes \alpha$. To see this, suppose $\iota : I \to J$ is an iso, with inverse κ. Since $\kappa\iota = \mathsf{id}_I$ and $\iota\kappa = \mathsf{id}_J$, the support of ι must be empty, and its place graph and link graph must be bijections $\iota^{\mathsf{P}} : m \to m$ and $\iota^{\mathsf{L}} : X \to Y$, where $I = \langle m, X \rangle$ and $J = \langle m, Y \rangle$. In other words, $\iota = \langle \pi, \alpha \rangle$, where π is a permutation on m and α is a renaming; this is readily expressed as $\iota = \pi \otimes \alpha$.

The only other elementary bigraph is a *discrete ion* $\mathsf{K}_{\vec{x}} : 1 \to \langle 1, \{\vec{x}\} \rangle$, for any sequence $\vec{x} = x_0, \ldots, x_{k-1}$ of distinct names where $k = ar(\mathsf{K})$ (see Figure 11).

We now turn to the first of our normal forms. It depends on the following two important concepts.

Definition 4.2 (Prime and discrete). An interface is *prime* if it has unit width. It takes the form $\langle 1, X \rangle$. A bigraph $G : I \to J$ is *prime* if J is prime and I has no names. A bigraph is *discrete* if every link joins exactly one point to an outer name.

Thus, a discrete bigraph is open and has no idle names. A discrete ion is an instance of a prime discrete bigraph. More generally, we define a *discrete molecule* M to be a prime discrete bigraph having a single outermost node.

The top part of Figure 12 shows a bigraph G and its underlying discrete bigraph D. Note that D consists just of discrete ions in a topographical arrangement; by composing it with a linking we recover G.

Our proof of the existence of normal forms will use the following notions.

Definition 4.3 (Firm and automorphism). A bigraph is *firm* if every root has exactly one child and every open link has exactly one point. An *automorphism* for F is an isomorphism ι such that $F\iota = F$.

We then have a cancellation property in the presence of firm bigraphs.

Proposition 4.4 (Cancelling). In abstract bigraphs, let $FG = FG'$ where G and G' are firm. Then $\iota G = G'$, where ι is an automorphism for F.

We are now ready to establish formally the existence of a *discrete normal form* (DNF) for a bigraph. We identify four levels of structure as four forms of expression. Taken in reverse order, these forms represent the DNF of any bigraph G; at each level, the expression is unique up to certain isomorphisms. It may be helpful to read the theorem in conjunction with the example described later, and illustrated in Figure 12.

Notation. In the theorem we shall use G to range over bigraphs, D over discrete bigraphs, P over prime discrete bigraphs and M over discrete molecules. We understand permutations π both as placing bigraphs and in the conventional way. With our bigraph convention, $\pi(i)$ means 'the root to which π maps the site i'; but we also need the convention, when π is a permutation of the sequence $0, 1, \ldots, n-1$, that $\pi\langle i\rangle$ means 'the i^{th} element in the permuted sequence'. Thus $\pi\langle i\rangle = \pi^{-1}(i)$.

Theorem 4.5 (Discrete normal form).

1 A discrete molecule M may be uniquely expressed as $M = (\mathsf{K}_{\vec{x}} \otimes \mathsf{id}_Y)P$, where P is a discrete prime.

2 A discrete prime may be expressed as

$$P = (merge_{n+k} \otimes \mathsf{id}_Y)(\mathsf{id}_n \otimes M_0 \otimes \cdots \otimes M_{k-1})\pi,$$

where each $M_i : m_i \to \langle i, Y_i\rangle$ is a discrete molecule. Any other such expression of P takes the form $(merge_{n+k} \otimes \mathsf{id}_Y)(\mathsf{id}_n \otimes M'_0 \otimes \cdots \otimes M'_{k-1})\pi'$, where there exist permutations ν on n, κ on k and μ_i on m_i $(i \in k)$ such that

$$M'_i = M_{\kappa\langle i\rangle}\mu_i \text{ and } (\nu \otimes \kappa')\pi = (\mathsf{id}_n \otimes \mu_0 \otimes \cdots \otimes \mu_{k-1})\pi',$$

where $\kappa' = \overline{\kappa}_{\vec{m}}$ is defined in terms of κ and $\vec{m}$ as in Lemma 5.4.

3 A discrete bigraph with outer width n may be expressed as

$$(P_0 \otimes \cdots \otimes P_{n-1})\pi \otimes \alpha,$$

where each P_i is prime and discrete. Any other such expression of D takes the form $(P'_0 \otimes \cdots \otimes P'_{n-1})\pi' \otimes \alpha$, where $P'_i = P_i\pi_i$ and $(\pi_0 \otimes \cdots \otimes \pi_{n-1})\pi' = \pi$ for certain permutations π_i.

4 A bigraph G with outer width n may be expressed as $(\mathsf{id}_n \otimes \omega)D$, where D is discrete. Any other such expression of G takes the form $(\mathsf{id}_n \otimes \omega')D'$, where $\omega' = \omega\alpha$ and $(\mathsf{id}_n \otimes \alpha)D' = D$ for some renaming α.

Proof. We shall outline the proof of (2); the others are easier. It is routine to confirm that an expression of the required form exists. If there are two such expressions, they must have the same n, k and Y; so we may write them as

$$P = (merge_{n+k} \otimes \mathsf{id}_Y)Q \quad \text{where} \quad Q = (\mathsf{id}_n \otimes M_0 \otimes \cdots \otimes M_{k-1})\pi$$
$$P = (merge_{n+k} \otimes \mathsf{id}_Y)Q' \quad \text{where} \quad Q' = (\mathsf{id}_n \otimes M'_0 \otimes \cdots \otimes M'_{k-1})\pi'.$$

Now Q and Q' are firm, so $\iota Q = Q'$ where ι is an automorphism for $merge_{n+k} \otimes \mathsf{id}_Y$. So ι takes the form $\rho \otimes \mathsf{id}_Y$; and structural considerations show that in this case $\rho = \nu \otimes \kappa$, where ν and κ permute n and k, respectively. Moreover, κ can be 'pushed through' $\vec{M}$ as in Lemma 5.4, and if we define $\kappa' = \overline{\kappa}_{\vec{m}}$ as in that lemma, we find

$$(\mathsf{id}_n \otimes M_{\kappa\langle 0\rangle} \otimes \cdots \otimes M_{\kappa\langle k-1\rangle})(\nu \otimes \kappa') = (\mathsf{id}_n \otimes M'_0 \otimes \cdots \otimes M'_{k-1})\pi'.$$

It follows by consideration of place graphs that $M'_i = M_{\kappa\langle i\rangle}\pi_i$ for some permutation π_i $(i \in k)$; thus, setting $R = \mathsf{id}_n \otimes M_{\kappa\langle 0\rangle} \otimes \cdots \otimes M_{\kappa\langle k-1\rangle}$, we deduce

$$R(\nu \otimes \kappa')\pi = R(\mathsf{id}_n \otimes \pi_0 \otimes \cdots \otimes \pi_{k-1})\pi'.$$

Now permutations are firm, so we can cancel R up to an automorphism ι, which, by consideration of place graphs, must take the form $\iota = \mathsf{id}_n \otimes \iota_0 \otimes \cdots \otimes \iota_{k-1}$ where $M_{\kappa\langle i\rangle}\iota_i = M_{\kappa\langle i\rangle}$. Now define $\mu_i \stackrel{\text{def}}{=} \iota_i\pi_i$. We can then conclude the proof; for we find $M'_i = M_{\kappa\langle i\rangle}\mu_i$, and by cancelling R we deduce

$$(\nu \otimes \kappa')\pi = (\mathsf{id}_n \otimes \mu_0 \otimes \cdots \otimes \mu_{k-1})\pi'. \qquad \square$$

This theorem reveals the placing and renaming isomorphisms that allow variation of expression in each of the four cases. For example, consider case (3) for $D : \langle m, X\rangle \to \langle n, Y\rangle$; each inner name $x \in X$ is linked to a distinct name $y \in Y$, which has no other children, and this fully determines the renaming α. However, the order of sites within each prime P_i may be chosen arbitrarily, and the product needs to be composed with a permutation correspondingly chosen to yield the correct order among all the sites of D.

Unique factorisation of a discrete bigraph D into primes justifies our definition of 'prime'; unicity would fail without the constraint that a prime has no inner names.

These DNFs are central to this paper. Of course, a bigraph can be expressed in many ways; but we shall later prove that, under certain algebraic laws, all other ways can be transformed into DNF. Thus the algebraic laws, being in this sense *complete*, give us a tight mathematical characterisation of bigraphs.

Example. Figure 12 shows part of the expression of a particular bigraph G in DNF. Using the equations of Theorem 4.5 in reverse order, we obtain in this case:

$$\begin{aligned} G &= (\mathsf{id}_1 \otimes \omega)D \qquad \text{where } \omega = {}^{x}/_{x_0x_1} \otimes {}^{y}/_{y_0y_1} \otimes ({}/_{z \circ z}/_{z_0z_1}) \\ D &= (P \otimes Q)\pi \otimes \alpha \quad \text{where } \alpha = {}^{x_1}/_{w} \text{ and } \pi : i \mapsto i - 1 (\mathrm{mod}\, 3) \\ P &= (merge_1 \otimes \mathsf{id}_{x_0y_0z_0})(\mathsf{id}_0 \otimes M_0)\mathsf{id}_1 = M_0 \\ M_0 &= (\mathsf{A}_{y_0} \otimes \mathsf{id}_{x_0z_0})P_0. \end{aligned}$$

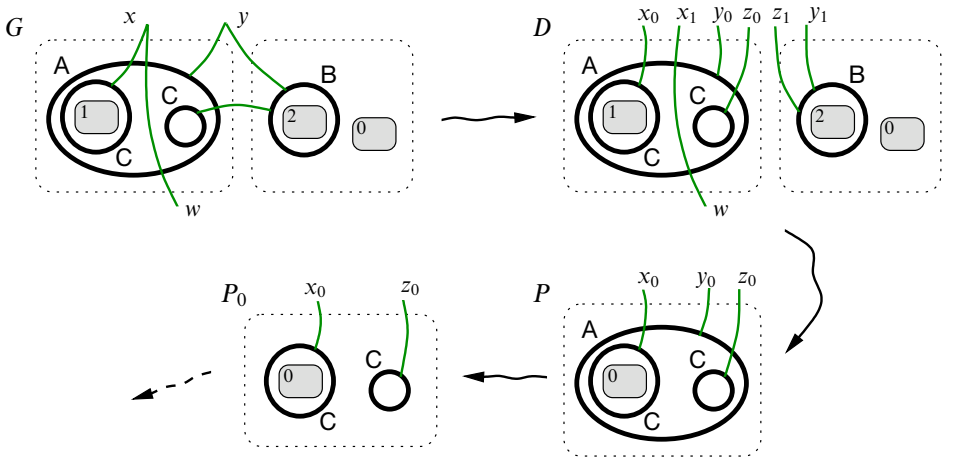


Fig. 12. Expressing a bigraph in discrete normal form.

The figure traces the analysis through the prime P, but not Q. It also shows P_0, the contents of A. The reader may care to resolve P_0 further; in fact, it simplifies to

$$P_0 = (join \otimes \mathsf{id}_{x_0 z_0})(\mathsf{C}_{x_0} \otimes \mathsf{C}_{z_0} 1) \ ;$$

note that the second molecule contains the empty region 1.

The reader may rightly criticise the verbosity of the expressions in this example, particularly the last one! Our DNF is not designed for programming convenience but for analysis. In Section 6 we briefly explore an alternative normal form, *connected* normal form or CNF, which employs *prime parallel product* '|' (parallel composition in CCS). This operation combines tensor product with merging and sharing of names. Here we can use it to write more succinctly $P_0 = \mathsf{C}_{x_0} \mid \mathsf{C}_{z_0} 1$. But, as we shall see in Section 6, there appears to be no easy axiomatisation that uses such a parallel product in place of the tensor product.

5. Axioms and completeness

We now address the question: What set of axioms is complete for equations between bigraph expressions, in the sense that every valid equation is provable? To answer this question, we must define our class of expressions precisely. We are considering expressions built by composition and tensor product from the identities and the following six classes of elementary forms:

$$1 \qquad join \qquad \gamma_{m,n} \qquad /x \qquad {}^{y}/_{X} \qquad \mathsf{K}_{\vec{x}}$$

where the $\vec{x}$ are distinct. Each expression E has two interfaces; we write $E : I \to J$, where I and J take the form $\langle m, X \rangle$, recalling that we allow abbreviations when $m = 0$ or $X = \varnothing$. In forming expressions, the interfaces for each composite expression (composition or tensor product) are determined from those of its components in the standard way. Then,

Table 1. *Axioms for bigraph equality.*

CATEGORICAL AXIOMS:

$$\begin{aligned}
A\,\mathsf{id} &= A = \mathsf{id}\,A \\
A(BC) &= (AB)C \\
A \otimes \mathsf{id}_\epsilon &= A = \mathsf{id}_\epsilon \otimes A \\
A \otimes (B \otimes C) &= (A \otimes B) \otimes C \\
\mathsf{id}_I \otimes \mathsf{id}_J &= \mathsf{id}_{I \otimes J} \\
(A_1 A_0) \otimes (B_1 B_0) &= (A_1 \otimes B_1)(A_0 \otimes B_0) \\
\gamma_{I,\epsilon} &= \mathsf{id}_I \\
\gamma_{J,I}\gamma_{I,J} &= \mathsf{id}_{I \otimes J} \\
\gamma_{I,K}(A \otimes B) &= (B \otimes A)\gamma_{H,J} \qquad (A : H \to I, B : J \to K)
\end{aligned}$$

LINK AXIOMS:

$$\begin{aligned}
x/x &= \mathsf{id}_x \\
{}^z/(Y \uplus y) \circ (\mathsf{id}_Y \otimes {}^y/X) &= {}^z/(Y \uplus X) \\
/y \circ {}^y/x &= /x \\
/y \circ y &= \mathsf{id}_\epsilon
\end{aligned}$$

PLACE AXIOMS:

$$\begin{aligned}
\mathit{join}(1 \otimes \mathsf{id}_1) &= \mathsf{id}_1 && \text{(unit)} \\
\mathit{join}(\mathit{join} \otimes \mathsf{id}_1) &= \mathit{join}(\mathsf{id}_1 \otimes \mathit{join}) && \text{(associative)} \\
\mathit{join}\gamma_{1,1} &= \mathit{join} && \text{(commutative)}
\end{aligned}$$

NODE AXIOMS:

$$(\mathsf{id}_1 \otimes \alpha)\mathsf{K}_{\tilde{x}} = \mathsf{K}_{\alpha(\tilde{x})}.$$

for any given expression E it is clear from the previous sections exactly which bigraph it denotes. As is standard, we write $\models E = F$ when the equation $E = F$ is *valid*, that is, when the expressions denote the same bigraph.

In formulating our axioms over expressions it is convenient to extend the symmetry expressions $\gamma_{m,n}$ to arbitrary interfaces by defining the following abbreviations:

$$\gamma_{I,J} \stackrel{\text{def}}{=} \gamma_{m,n} \otimes \mathsf{id}_{X \uplus Y} \quad \text{where } I = \langle m, X \rangle, J = \langle n, Y \rangle.$$

With the help of these, we define the equational axioms in Table 1. As is standard, we shall write $\vdash E = F$ to mean that the equation is *provable*, that is, can be deduced from the axioms. The rest of this section is devoted to answering the opening question of this section; indeed, we shall prove that $\vdash E = F$ if and only if $\models E = F$. The forward implication means that our axioms are sound; the backwards implication means they are complete.

Before embarking on the proof, we note a few points about the axioms.

Categorical axioms: These are standard for a strict symmetric monoidal category. But note that the tensor product is defined only when interfaces have disjoint name sets;

thus the equations are required to hold only when both sides are defined. The two axioms $\vdash \mathsf{id}_I \otimes \mathsf{id}_J = \mathsf{id}_{I \otimes J}$ and $\vdash (A_1 A_0) \otimes (B_1 B_0) = (A_1 \otimes B_1)(A_0 \otimes B_0)$ express the fact that the tensor product is bifunctorial.

The last three categorical axioms are concerned with symmetries. Note that from the final axiom we can prove $\vdash A \otimes B = B \otimes A$ for link graphs. Indeed, if $H = X$ and $I = Y$, we have $\vdash \gamma_{H,I} = \gamma_{\epsilon,\epsilon} \otimes \mathsf{id}_{X \uplus Y} = \mathsf{id}_{X \uplus Y}$ by definition, using also the axioms $\vdash \gamma_{\epsilon,\epsilon} = \mathsf{id}_\epsilon$ and $\vdash \mathsf{id}_\epsilon \otimes A = A$; a similar result for $\gamma_{J,K}$ completes the proof.

Link axioms: The first two of these axioms express obvious properties of substitutions. The last two are novel, because closure $/x$ is not normally studied; remarkably, they tell us all we need to know about closure. For example, we can deduce a form of alpha-conversion for closure, namely $\vdash /z \circ {}^z/X = /w \circ {}^w/X$. To see this, first put $Y = \varnothing$ and $y = w$ in the second link axiom to get $\vdash {}^z/w \circ {}^w/X = {}^z/X$, then use this in combination with the third link axiom.

Place axioms: These axioms are intuitive and need little explanation. They do not tell the whole story about placing; the three categorical axioms for symmetries are important for the permutation of places.

Node axioms: No axioms are required on ions except this simple renaming axiom, and this is needed only because names are treated positionally. This indicates that bigraphs are a rather free structure.

The soundness of our axiom system, that is, that $\vdash E = F$ implies $\models E = F$, is straightforward. The validity of each axiom follows routinely from the definition of the categorical operations as given in Section 3, and the definition of the placings, linkings and ions as given in Section 4.

We now turn to the main task, to prove completeness.

Preliminaries

In this subsection we prove some useful lemmas, including completeness for certain subclasses of expressions. We say that a class is *generated* by a set of elementary expressions if it is the smallest class including those elementary expressions and closed under composition and tensor product. In particular: the *permutation expressions* are generated by symmetries $\gamma_{m,n}$ and place identities; the *place expressions* are generated by these together with 1 and *join*; and the *link expressions* are generated by closures $/x$, substitutions ${}^y/X$ and link identities.

We now state without proof a standard result of symmetric monoidal categories.

Proposition 5.1 (Permutation completeness). The theory is complete for permutation expressions.

To extend this to place expressions, we need some auxiliary results. They follow from the axioms, together with the inductive definition of $merge_m$ (Definition 4.1).

Lemma 5.2.

1 $\vdash merge_1 = join(\mathsf{id}_1 \otimes 1) = \mathsf{id}_1$.

2 $\vdash merge_{m+n} = join(merge_m \otimes merge_n)$.

3 $\vdash merge_{m+1} = join(merge_m \otimes \mathsf{id}_1)$.
4 $\vdash merge_{m+1}\gamma_{m,1} = merge_{m+1}$.

Proof (*outline*). The first part just requires manipulation of the axioms; the second and third require the first, with induction on m; the fourth requires the third. □

Lemma 5.3.

1 $\vdash merge_m\pi = merge_m$.
2 $\vdash merge_m(merge_{n_0} \otimes \cdots \otimes merge_{n_{m-1}}) = merge_n$, where $n = \sum_{i\in m} n_i$.

Proof (*outline*). The first part is proved by induction on m. The inductive step requires the fact that any permutation π on $m{+}1$ can be expressed as $\pi = (\mathsf{id}_1 \otimes \pi')(\gamma_{m-k,1} \otimes \mathsf{id}_k)$, where π' is a permutation on m and $0 \leqslant k \leqslant m$. More strongly, by the completeness for permutation expressions, we can assert that this equation is provable in the theory.

The second part requires Lemma 5.2(2), together with induction on m. □

We need a further property of permutations. The axiom $\vdash \gamma_{I,K}(A \otimes B) = (B \otimes A)\gamma_{H,J}$ 'pushes' the elementary permutation $\gamma_{I,K}$ through a binary product. We need to generalise this to allow a permutation in n to be pushed through a product of n primes. Moreover, this property must be provable from the axioms.

Lemma 5.4. Let π be a permutation expression on n, and let $\vec{m} = m_0, \ldots m_{n-1}$ be finite ordinals. Then there exists a permutation expression $\overline{\pi}_{\vec{m}}$ such that, for any n prime expressions $P_i : m_i \to \langle 1, X_i\rangle$ for $i \in n$, with $X = \biguplus_{i\in n} X_i$,

$$\vdash (\pi \otimes \mathsf{id}_X)(P_0 \otimes \cdots \otimes P_{n-1}) = (P_{\pi\langle 0\rangle} \otimes \cdots \otimes P_{\pi\langle n-1\rangle})\overline{\pi}_{\vec{m}}.$$

Proof. We proceed by induction on n. The basis $n = 0$ is trivial, since we must have $\vdash \pi = \mathsf{id}_0$ and we choose $\overline{\pi}$ to be id_0.

For the inductive step, let ρ be a permutation expression on $n{+}1$. We can consider the permutation ρ as first applying a permutation π on the first n of $n{+}1$ elements, then inserting the last of them just after the first k in the permuted sequence, for some $k \in n{+}1$. So, knowing completeness for permutation expressions, we have

$$\vdash \rho = (\mathsf{id}_k \otimes \gamma_{n-k,1})(\pi \otimes \mathsf{id}_1).$$

Thus $\rho\langle i\rangle = \pi\langle i\rangle$ for $i < k$, $\rho\langle k\rangle = m_n$, and $\rho\langle i\rangle = \pi\langle i{-}1\rangle$ for $i > k$. Let $\vec{m}$ and P_i $(i \in n)$ be as in the lemma statement, with an additional prime expression $P_n : m_n \to \langle 1, X_n\rangle$. We have to define $\overline{\rho}_{\vec{m},m_n}$ so that

$$\vdash (\rho \otimes \mathsf{id}_{X\uplus X_n})(P_0 \otimes \cdots \otimes P_n) = (P_{\rho\langle 0\rangle} \otimes \cdots \otimes P_{\rho\langle n\rangle})\overline{\rho}_{\vec{m},m_n}. \qquad (*)$$

For this purpose, let $\ell = \sum_{i\in k} m_{\pi(i)}$, the total inner width of $P_{\pi\langle 0\rangle}, \ldots, P_{\pi\langle k-1\rangle}$; then $m{-}\ell$ is the total inner width of $P_{\pi\langle k\rangle}, \ldots, P_{\pi\langle n-1\rangle}$. Define

$$\overline{\rho}_{\vec{m},m_n} \stackrel{\text{def}}{=} (\mathsf{id}_\ell \otimes \gamma_{m-\ell,m_n})(\overline{\pi}_{\vec{m}} \otimes \mathsf{id}_{m_n}).$$

Then, by routine manipulation, using the axioms and the inductive assumption, we can verify the required property $(*)$ for $\rho_{\vec{m},m_n}$, and we are done. □

We are now ready to prove some special instances of completeness.

Proposition 5.5 (Place completeness). The theory is complete for place expressions.

Proof. First we show for every place expression E that

$$\vdash E = (merge_{m_0} \otimes \cdots \otimes merge_{m_{k-1}})\pi$$

for some $k \geqslant 0$ and permutation expression π. The proof is by structural induction on expressions. For the inductive step, if it holds for E and F, it immediately holds for $E \otimes F$, and to show it for EF amounts to a simple use of Lemmas 5.3 and 5.4.

Now suppose F is another place expression with $\models E = F$ and

$$\vdash F = (merge_{n_0} \otimes \cdots \otimes merge_{n_{\ell-1}})\pi'.$$

Then we must have $k = \ell$, $m_i = n_i$, and $\models \pi' = (\rho_0 \otimes \cdots \otimes \rho_{k-1})\pi$ for some ρ_i. Hence, also, $\vdash \pi' = (\rho_0 \otimes \cdots \otimes \rho_{k-1})\pi$, by permutation completeness. We also have $\vdash merge_{m_i}\rho_i = merge_{m_i}$ from Lemma 5.3, so we deduce $\vdash E = F$, as required. □

Proposition 5.6 (Link completeness). The theory is complete for link expressions.

Proof. First we show for every link expression E that $\vdash E = \widehat{E}$, where $\widehat{E}$ is a *link normal form*, which is either id_ϵ or a product of terms of the three kinds

$$/y \qquad {}^{x}/Y \qquad /x \circ {}^{x}/Y,$$

where in the second kind Y may be any set (including $\varnothing$), and in the third kind Y may be any set with at least two members. The latter kind represents a closed link between two or more inner names.

We now proceed by structural induction. We consider all cases for the form of E. If it is either $/x$ or ${}^{x}/Y$, we take $\widehat{E}$ to be E itself. If E is id_X with $X = \{x_0, \ldots, x_{n-1}\}$, then, from the axioms $\vdash \mathsf{id}_{X \otimes Y} = \mathsf{id}_X \otimes \mathsf{id}_Y$ and $\vdash {}^{x}/x = \mathsf{id}_x$, we derive

$$\vdash E = \mathsf{id}_{x_0} \otimes \cdots \otimes \mathsf{id}_{x_{n-1}} = {}^{x_0}/x_0 \otimes \cdots \otimes {}^{x_{n-1}}/x_{n-1}.$$

If E is a tensor product $E_1 \otimes E_2$, the result is obvious by induction.

The last possibility is that E is a composition E_1E_2. Then, by induction, $\vdash E = \widehat{E_1}\widehat{E_2}$. One situation that arises when converting this by the axioms into a normal form is when $\widehat{E_1}$ contains a term ${}^{x}/Y$ with $Y = \{y_0, \ldots, y_{n-1}\}$, and $\widehat{E_2}$ contains n terms ${}^{y_i}/Z_i$ for $i \in n$. Then, using categorical axioms, $\widehat{E_1}\widehat{E_2}$ can be provably converted to a tensor product containing the term

$${}^{x}/Y \circ ({}^{y_0}/Z_0 \otimes \cdots \otimes {}^{y_{n-1}}/Z_{n-1}),$$

and, by repeated use of the second link axiom, this can be brought into the form ${}^{x}/Z$ where $Z = Z_0 \uplus \cdots \uplus Z_{n-1}$.

A similar situation arises when $\widehat{E_1}$ contains a term $/x \circ {}^{x}/Y$, and it can be treated similarly. We leave the reader to supply the details. This completes the inductive proof that $\vdash E = \widehat{E}$ for some link normal form $\widehat{E}$.

Now, by soundness, if $\models E = F$, then $\models \widehat{E} = \widehat{F}$ also. But our normal form is such that two of them denoting the same bigraph must contain exactly the same terms, though possibly ordered differently. So $\vdash \widehat{E} = \widehat{F}$, hence $\vdash E = F$, as required. □

We now approach the completeness of the theory for bigraphs. In order to reduce it to completeness for discrete bigraphs, we need a syntactic version of DNF. In particular, we shall show that any discrete bigraph has a linear expression, defined as follows.

Definition 5.7 (Linear). A bigraph expression is *linear* if it contains no restrictions, and no substitution elements except *linear* ones – those of the form $^{y}/_{x}$.

As we shall see in Proposition 5.13, a linear expression always denotes a discrete bigraph. This is intuitively clear; by banning $^{y}/_{X}$ except when X is a singleton, we ensure all links contain exactly one point, and by banning $/x$ we exclude closed links.

Clearly, all sub-expressions of a linear expression are linear; so linearity is amenable to structural induction. For example, we have the following lemma.

Lemma 5.8. If E is linear, $\vdash E = E' \otimes \alpha$ where E' is linear without inner names.

Proof. We use structural induction. The proof for elements and identities is easy; so is the inductive step for product. It remains to show the inductive step for composition, that is, if the property holds for E and F, it holds for EF. Let EF be linear with $\vdash E = E' \otimes \alpha$ and $\vdash F = F' \otimes \beta$, where E' and F' have no inner names. We have $\vdash \alpha = \alpha_0 \otimes \alpha_1$, where the domains of α_0, α_1 are the outer names of F and β, respectively. Then $\vdash EF = (E' \otimes \alpha_0)F' \otimes \alpha_1\beta$, which is of the required form. □

We end this subsection with a first approximation to a provable normal form for arbitrary bigraph expressions.

Proposition 5.9 (Underlying linear expression). For any expression G of outer width m there exist ω and linear E such that $\vdash G = (\mathsf{id}_m \otimes \omega)E$.

Proof. We use structural induction again, and we need only look at the inductive step for composition. Consider GH, and assume that for some linear E and F

$$\vdash G = (\mathsf{id}_m \otimes \omega)E \quad \text{and} \quad \vdash H = (\mathsf{id}_n \otimes \omega')F.$$

Then, by Lemma 5.8, we have $\vdash E = E' \otimes \alpha$ where E' has no inner names. Hence

$$\begin{aligned} \vdash GH &= (\mathsf{id}_m \otimes \omega)(E' \otimes \alpha\omega')F \\ &= (\mathsf{id}_m \otimes \omega(\mathsf{id}_Y \otimes \alpha\omega'))(E' \otimes \mathsf{id}_X)F, \end{aligned}$$

where X and Y are the outer names of F and E', respectively. This is of the required form, since $(E' \otimes \mathsf{id}_X)F$ is linear. □

Provable normal forms

We are interested in four increasingly general kinds of expression: those that denote discrete molecules, discrete primes, discrete bigraphs and arbitrary bigraphs, respectively. We now set out to prove that every expression of each kind can be proved equal to the corresponding kind of DNF, defined as follows (as suggested in the previous section).

Definition 5.10 (Discrete normal forms). There are four kinds of DNF: an MDNF M for discrete molecules, a PDNF P for discrete primes, a DDNF D for discrete bigraphs and a

BDNF B for bigraphs:

$$\begin{aligned}
&\text{MDNF } M ::= (\mathsf{K}_{\hat{x}} \otimes \mathsf{id}_Y)P\\
&\text{PDNF } P ::= (merge_{n+k} \otimes \mathsf{id}_Y)(\mathsf{id}_n \otimes M_0 \otimes \cdots \otimes M_{k-1})\pi\\
&\text{DDNF } D ::= (P_0 \otimes \cdots \otimes P_{n-1})\pi \otimes \alpha\\
&\text{BDNF } B ::= (\mathsf{id}_n \otimes \omega)D\ .
\end{aligned}$$

We begin with a lemma showing that normal forms are provably closed under isomorphism, in a certain sense.

Lemma 5.11. Let $B : I \to I'$ be a BDNF. If $\imath$ and $\imath'$ are isomorphisms on I and I', respectively, then $\vdash \imath' B \imath = B'$ for some BDNF B'.

The analogous property holds also for DDNF, PDNF and MDNF.

We omit the proof, which depends on Lemma 5.4; it uses induction on the number of ions in an expression. Our next lemma uses a more complex instance of this technique, and shows that DNFs are provably closed under certain compositions.

Lemma 5.12. Let $C : \ell \to \langle m, Z \rangle$ be a product of PDNFs. Then:

1 If $M : m \to \langle 1, Y \rangle$ is an MDNF, then $\vdash (M \otimes \mathsf{id}_Z)C = M'$ for some MDNF M'.
2 If $P : m \to \langle 1, Y \rangle$ is a PDNF, then $\vdash (P \otimes \mathsf{id}_Z)C = P'$ for some PDNF P'.
3 If $D : m \to \langle n, Y \rangle$ is a DDNF, then $\vdash DC = D'$ for some DDNF D'.

Proof. We first prove (1) and (2) by simultaneous induction on the number n of ions in P or M. Assume both parts hold for $< n$ ions.

For (1) with n ions, let M be $(\mathsf{K}_{\hat{x}} \otimes \mathsf{id}_W)P$ for PDNF $P : m \to \langle 1, W \rangle$. Then

$$\vdash (M \otimes \mathsf{id}_Z)C = (\mathsf{K}_{\hat{x}} \otimes \mathsf{id}_{W \uplus Z})(P \otimes \mathsf{id}_Z)C,$$

where P has $n-1$ ions. We may therefore apply (2) to P, yielding the required result.

For (2) with n ions, let P be $(merge_{h+k} \otimes \mathsf{id}_Y)(\mathsf{id}_h \otimes M_0 \otimes \cdots \otimes M_{k-1})\pi$. Now C is a product $Q_0 \otimes \cdots \otimes Q_{m-1}$ of PDNFs, and by Lemma 5.4, we may push π through it to get, for some π',

$$\vdash (P \otimes \mathsf{id}_Z)C = (merge_{h+k} \otimes \mathsf{id}_{Y \uplus Z})(\mathsf{id}_{\langle h,Z \rangle} \otimes \bigotimes_{i \in k} M_i)(Q_{\pi\langle 0 \rangle} \otimes \cdots \otimes Q_{\pi\langle m-1 \rangle})\pi'.$$

Now the sequence of Q_j can be factored into products $C', C_0 \ldots, C_{k-1}$ with outer names $Z', Z_0, \ldots Z_{k-1}$, where $Z' \uplus \biguplus_i Z_i = Z$, yielding

$$\vdash (P \otimes \mathsf{id}_Z)C = (merge_{h+k} \otimes \mathsf{id}_{Y \uplus Z})\big(C' \otimes \bigotimes_{i \in k}(M_i \otimes \mathsf{id}_{Z_i})C_i\big)\pi'.$$

Since each M_i has at most n ions, we can apply (1) to each $(M_i \otimes \mathsf{id}_{Z_i})C_i$. Furthermore, using Lemma 5.3(2), the merge in each prime factor of C' may be combined with the outer merge. Together, these manipulations provably yield a DPNF P', as required. This concludes the inductive proof of (1) and (2).

To prove (3), note that D takes the form $(P_0 \otimes \cdots \otimes P_{n-1})\pi \otimes \alpha$. We first push π through C by Lemma 5.4. Then we apply (2) to n forms $(P_i \otimes \mathsf{id}_{Z_i})C_i$. The result is then provably equal to a DDNF D' using Lemma 5.11. □

We are now ready for our main result on provable normal forms.

Proposition 5.13 (Provable normal forms). Let E be a linear expression.

1 If E denotes a molecule, then $\vdash E = M$ for some MDNF M.
2 If E denotes a prime, then $\vdash E = P$ for some PDNF P.
3 If E denotes a bigraph, then $\vdash E = D$ for some DDNF D.
4 If G is any expression, then $\vdash G = B$ for some BDNF B.

Proof. We first prove (3) by structural induction. The base case (elementary linear expressions) and the case for tensor product of two linear expressions are straightforward. For the case of a composition, suppose the result holds for linear E_0 and E_1; we wish to prove it for E_0E_1. By assumption, we have $\vdash E_i = D_i$ for DDNF D_i $(i = 0, 1)$. Now D_1 has the form $C\pi \otimes \beta$, where C is a product of PDNFs, and we have $\vdash D_0 = D_0' \otimes \alpha$ where D_0' is DDNF and α is composable with β. Hence

$$\vdash E_0E_1 = D_0'C\pi \otimes \alpha\beta = D \otimes \alpha\beta$$

where D is a DDNF by Lemma 5.12, so we are done.

For (2), we first note from (1) that $\vdash E = D$, which is a DDNF. But since D has outer width 1 and no inner names, by inspection of the DDNF structure we find (with the help of Lemma 5.11) that $\vdash D = P$, which is a PDNF, and we are done.

For (3), we first note from (2) that $\vdash E = P$, which is a PDNF. But since P denotes a molecule, as in the previous case we find that $\vdash D = M$, which is a MDNF, and we are done.

Finally, (4) follows directly from (3) and Proposition 5.9. □

Completeness

We are now ready to prove completeness. We first need an inductive argument to prove it for linear expressions, and full completeness then follows directly.

Proposition 5.14 (Linear completeness). If E and E' are linear expressions and $\models E = E'$, then $\vdash E = E'$.

Proof. We first prove completeness for prime linear expressions by induction on the number of ions in E (and hence also in E'). Assume the result holds for $< n$ ions.

First we prove the result when E and E', with n ions, denote a molecule. In this case, by Proposition 5.13(1) and Theorem 4.5(1), we have provable MDNFs

$$\begin{aligned} \vdash E &= (\mathsf{K}_{\vec{x}} \otimes \mathsf{id}_Y)P \\ \vdash E' &= (\mathsf{K}_{\vec{x}} \otimes \mathsf{id}_Y)P' \end{aligned}$$

where P and P' are PDNFs with $< n$ ions such that $\models P = P'$. By the induction hypothesis, we have $\vdash P = P'$, and it follows that $\vdash E = E'$.

Now we extend the result to when E and E', with n ions, denote any prime. By Proposition 5.13(2) and Theorem 4.5(2), we have provable PDNFs

$$\begin{aligned} \vdash E &= (merge_{n+k} \otimes \mathsf{id}_Y)(\mathsf{id}_n \otimes M_0 \otimes \cdots \otimes M_{k-1})\pi \\ \vdash E' &= (merge_{n+k} \otimes \mathsf{id}_Y)(\mathsf{id}_n \otimes M_0' \otimes \cdots \otimes M_{k-1}')\pi' \end{aligned}$$

where M_i, M_i', π and π' satisfy the conditions set out in Theorem 4.5(2). But these discrete molecules have no more than n ions, so we apply the above result to obtain provable equality of certain MDNFs and then, with the help of Lemma 5.3(1), provable equality for the two displayed PDNFs; hence again $\vdash E = E'$. This completes the inductive proof of the proposition for prime linear expressions.

Finally, suppose that E and E' are any linear expressions. By Proposition 5.13(3) and Theorem 4.5(3), we have provable DDNFs

$$\begin{aligned} \vdash E &= (P_0 \otimes \cdots \otimes P_{n-1})\pi \otimes \alpha \\ \vdash E' &= (P_0' \otimes \cdots \otimes P_{n-1}')\pi' \otimes \alpha', \end{aligned}$$

where the P_i and P_i' are PDNFs such that $\models P_i' = P_i\rho_i$ and $\models (\rho_0 \otimes \cdots \otimes \rho_{n-1})\pi' = \pi$ for certain permutations ρ_i. But, with the help of Lemma 5.11, our inductive argument shows that the first equation is provable, and the second is also provable by Proposition 5.5. It follows immediately that $\vdash E = E'$. □

Theorem 5.15 (Completeness). If $\models G = G'$, then $\vdash G = G'$.

Proof. By Proposition 5.13(4) and Theorem 4.5(4), we have provable DDNFs

$$\begin{aligned} \vdash G &= (\mathsf{id}_n \otimes \omega)E \\ \vdash G' &= (\mathsf{id}_n \otimes \omega')E', \end{aligned}$$

where E and E' are DDNFs such that $\models E = (\mathsf{id}_n \otimes \alpha)E'$ and $\models \omega\alpha = \omega'$ for some renaming α. But Proposition 5.14, with Lemma 5.11, shows that the first equation is provable, and the second is also provable by Proposition 5.6. It follows that $\vdash G = G'$. □

6. Programming and connected normal form

The completeness of the axiom system in Table 1 depends primarily on two things: first, that all linking can be exposed at the outermost level of an expression; second, that we have a strict symmetric monoidal category of bigraphs, with a tensor that is partial on objects. Crucial to the tensor is that it is bifunctorial, that is, $(A_1A_0) \otimes (B_1B_0) = (A_1 \otimes B_1)(A_0 \otimes B_0)$; this axiom underlies most of our manipulations.

Thus the discrete normal form, DNF, has been crucial for the proof of completeness. It also gives great insight into the bigraph model, by revealing the orthogonality between placing and linking. In this section we briefly discuss another normal form for bigraphs. One purpose is to underline the primacy of DNF due to its mathematical role in the theory; but we also wish to point out that another normal form exists (which is not so common in algebraic theories), and that it has certain practical advantages.

Despite its mathematical significance, DNF is not very convenient for programming, or more generally for the practical description of systems. As a striking example of this, let us consider the calculus of mobile ambients (Cardelli and Gordon 2000), which can be formulated within bigraphs. Ambients are regions within which local activity may occur, and their mobility is captured by certain reaction rules. We have already illustrated one of them in Figure 6.

In this paper we are not concerned with the dynamics, but only with the statics of bigraphs. But we can use the redex R of this reaction rule just as an example of a realistic bigraph. Let us write down a slightly simplified version of what R would look like in DNF:

$$R = (merge \otimes \mathsf{id}_x \otimes y/yz)\big((\mathsf{amb}_x \otimes \mathsf{id}_y)merge(\mathsf{id}_1 \otimes \mathsf{in}_y 1) \otimes \mathsf{amb}_z\big).$$

Compare this with the equivalent expression

$$R = \mathsf{amb}_x(\mathsf{id}_1 \mid \mathsf{in}_y 1) \mid \mathsf{amb}_y,$$

which, with a little sugaring, corresponds to the standard notation of the ambient calculus. Instead of DNF it uses the *connected normal form* (CNF), which we shall shortly define in terms of the *prime parallel product* '|', which combines tensor product, merging and substitution. CNF resembles parallel composition in other process calculi; for example, the (only) reaction rule of a simplified π-calculus takes the form

$$\overline{x}y.P \mid x(z).Q \longrightarrow P \mid \{y/z\}Q.$$

Here, a sender and receiver are placed side-by-side (compare tensor product) in the same region (compare *merge*) and sharing the channel name x (compare substitution).

Parallel product

We now formally define the operation of *parallel product* '$\parallel$' on bigraphs of arbitrary width. It resembles tensor product, but allows names to be shared. Thus, on interfaces it is always defined:

$$\langle m, X\rangle \parallel \langle n, Y\rangle \stackrel{\text{def}}{=} \langle m+n, X \cup Y\rangle.$$

On bigraphs with disjoint inner names it is always defined. Suppose that $X_0 \cap X_1 = \varnothing$, and let $G_i : \langle m_i, X_i\rangle \to \langle n_i, Y_i\rangle$ for $i = 0, 1$. Then we define the parallel product of G_0 and G_1 in terms of tensor product. First we disjoin their outer names by applying renamings $\alpha_i : Y_i \to Y_i'$ with $Y_0' \cap Y_1' = \varnothing$. Then, setting $\sigma = \alpha_0^{-1} \cup \alpha_1^{-1}$, we define

$$G_0 \parallel G_1 \stackrel{\text{def}}{=} \sigma(\alpha_0 G_0 \otimes \alpha_1 G_1) \quad : \langle m_0+m_1, X_0 \uplus X_1\rangle \to \langle n_0+n_1, Y_0 \cup Y_1\rangle.$$

Just like $\otimes$, this product is associative, with unit id_ϵ. It is defined more often than $\otimes$, and when both products are defined, they are equal. But, unlike $\otimes$, it is not bifunctorial; instead it satisfies

$$(F_0 \parallel F_1)(G_0 \otimes G_1) \;=\; (F_0 G_0) \parallel (F_1 G_1),$$

which suggests that, algebraically, $\parallel$ does not stand well by itself; it needs $\otimes$ to be present as an auxiliary.

Before defining CNF, we need another operator, the *prime parallel product* '|', which always creates a prime, even on non-prime arguments. (In contrast, if G_0 and G_1 are prime, $G_0 \parallel G_1$ has width 2). It is defined simply by

$$G_0 \mid G_1 \;\stackrel{\text{def}}{=}\; merge_n(G_0 \parallel G_1).$$

It is again associative; on primes it has the prime 1 as a unit. Again, it is not bifunctorial but satisfies

$$(F_0 \mid F_1)(G_0 \otimes G_1) \;=\; (F_0 G_0) \mid (F_1 G_1).$$

This operator agrees strongly with operators in process calculi; indeed, when we translate either the π-calculus or the calculus of mobile ambients into bigraphs, '$\mid$' in the calculus is encoded by '$\mid$' in bigraphs. The difference is that those calculi do not in general have categorical composition; their processes correspond to those bigraphs whose inner face is ϵ. However, the dynamic theory of bigraphs (Jensen and Milner 2004) allows us to *derive* labelled transition systems where, in each transition $G \xrightarrow{L} G'$, the label L is itself a bigraph (normally a little one) such that $L \circ G$ is defined. Treating labels as composable entities is essential to the uniform behavioural theory that bigraphs provide.

Connected normal form

The key distinction of the parallel products is that they allow sharing of names. This sharing is induced by the substitution σ used in their definition. Thus, by using them instead of tensor product, we are pushing substitutions *inwards* as far as possible. The key to connected normal form, then, is to push *all* linking inwards as far as possible, including closures. This is a close parallel to the good advice often given to programmers to declare their variables in as small a scope as possible. With this in mind, we are ready for our final definition.

Definition 6.1 (Connected normal forms). There are three kinds of CNF: MCNFs (M for molecules), PCNFs (P for primes) and BCNFs (B for bigraphs):

$$\begin{aligned}
&\text{MCNF: } M ::= (/Z \mid \mathsf{id}_1)(\mathsf{K}_{\vec{x}} \mid \mathsf{id}_Y)P \\
&\text{PCNF: } P ::= (/Z \mid \mathsf{id}_1)(\mathsf{id}_n \mid M_0 \mid \cdots \mid M_{k-1})\pi \\
&\text{BCNF: } B ::= (/Z \parallel \mathsf{id}_n)(\sigma \parallel (P_0 \parallel \cdots \parallel P_{n-1})\pi)\ .
\end{aligned}$$

The names $\vec{x}$ need not be distinct in the MCNF. Moreover, in each case, any closed name $z \in Z$ must occur in at least two members of the ensuing product ($\parallel$ or $\mid$).

Many of the technical properties of DNF are shared by CNF. First, every molecule (prime, bigraph, respectively) can be expressed by a MCNF (PCNF, BCNF, respectively). Second, these expressions are unique up to certain isomorphisms, which are close to those described in Theorem 4.5.

However, we leave open the question of a complete axiomatisation expressed in terms of the parallel products instead of the tensor product. Even if it exists, it is unlikely to be as simple, because the bifunctorial property of the tensor product is absent. This does not alter the fact that the CNFs appear to be easier to use in applications. I conjecture that a programming language for bigraphs may well be based firmly on CNFs but with added notational convenience, just as existing languages are based on process calculi. Moreover, the conversion to CNF is not hard, and can be done whenever necessary for theoretical analysis.

7. Related and further work

The algebraic formulation of bigraphs arises from a category in which the objects are interfaces and the arrows are graphs, following Lawvere's paradigm for algebraic theories (Lawvere 1963). It is interesting and non-trivial to explore the relationship between this treatment, particularly as it affects dynamic theory, with that of graph-rewriting using the double pushout construction (Ehrig 1979), where the objects are graphs and the arrows are graph embeddings. The Lawvere approach is closer to the algebraic tradition in process calculi; we are keen to adopt it both for comparison with these calculi and because the algebraic approach to concurrent processes has been fruitful. But a link between the two approaches has been identified (Ehrig 2002), and indeed the techniques for deriving behavioural congruences has been transferred (Ehrig and König 2004) from the algebraic framework to the embedding framework. More remains to be done to discover the relative benefits of the approaches, both in allowing different kinds of graph and in analysing real systems in practice.

Process calculi, extended with stochastics, are becoming successful in modelling biological processes, for example signal transduction in cells (Priami *et al.* 2001). Recently, as one might expect, explicitly topographic models such as mobile ambients are being employed in this way (Regev *et al.* 2004). This link with biology is encouraging, and also valuable as a test for the wider applicability of process models arising from computer science. Also, as our opening example suggests, there is a strong incentive to build topographical calculi that can model the phenomena of ubiquitous computing, where mobile communicating automata (both macro- and microscopic) will abound.

Let us now turn to possible variations and developments in bigraph theory. The present definition is not canonical. For example, for some purposes we may wish to consider bigraphs whose regions (nodes) may overlap one another. For example, one node may represent a physical location, say Los Angeles, and an overlapping node may represent the University of California, which has separate campuses in many Californian cities. Can bigraphs respond to this challenge? At one level, it is easy; we simply generalise place graphs from forests of trees to directed (acyclic) graphs. However, the impact of this change on our algebra is not obvious.

Another variation is to consider *polygraphs*, where there may be more than two orthogonal structures. Indeed, this may be the way to cope with overlapping nodes; for we may consider nodes to be structured both by nesting of *physical* regions (such as Los Angeles) and by nesting of *virtual* regions (such as the University of California), without either form of locality constraining the other. Indeed, the dynamic theory of bigraphs (Jensen and Milner 2004) does not exclude this, since it relies on categorical concepts developed separately for each of the orthogonal structures. But again, the impact on algebraic theory is not yet clear.

Within our present theory, much refinement is possible and is being examined. The first is to do with local names, or localised links. Once the theory is developed, one can constrain the independence of linking and placing by introducing *bound* names, those that confine a link to within a certain place. This is needed for modelling established process calculi, and it has turned out (Jensen and Milner 2003; Jensen and Milner 2004) that

the refined theory can readily be embedded in the pure theory. We conjecture that the present algebraic theory can be simply adapted to the refined theory. From the algebraic viewpoint, an even more obvious refinement is to introduce *sorting*, allowing both the regions and the names to be many-sorted. This has already been employed in modelling Petri nets (Milner 2004; Leifer and Milner 2004) and the full π-calculus (Jensen 2004).

To conclude, the algebraic treatment of mobile processes with explicit regions has considerably potential, and the purpose of the present paper has been to apply traditional algebraic methodology to the static structure of such processes, and thus to provide a firm basis for the supervening dynamical theory.

Acknowledgements

I would like to thank the anonymous referees for helping me to express many things more clearly.

References

Benson, D. B. (1975) The basic algebraic structures in categories of derivations. *Information and Control* **28** 1–29.

Cardelli, L. and Gordon, A. D. (2000) Mobile ambients. In: Foundations of System Specification and Computational Structures. *Springer-Verlag Lecture Notes in Computer Science* **1378** 140–155.

Cattani, G. L., Leifer, J. J. and Milner, R. (2000) Contexts and Embeddings for closed shallow action graphs. Technical Report 496, University of Cambridge Computer Laboratory. (Available at `http://pauillac.inria.fr/~leifer`.)

Ehrig, H. (1979) Introduction to the theory of graph grammars. In: Graph Grammars and their Application to Computer Science and Biology. *Springer-Verlag Lecture Notes in Computer Science* **73** 1–69.

Ehrig, H. (2002) Bigraphs meet double pushouts. *EATCS Bulletin* **78** 72–85.

Ehrig, H. and König, B. (2004) Deriving bisimulation congruences in the DPO approach to graph rewriting. In: Foundations of Software Science and Computation Structures, Proc. 7th International Conference. *Springer-Verlag Lecture Notes in Computer Science* **2987** 151–166.

Gardner, P. A. (2000) From process calculi to process frameworks. *Proc. CONCUR 2000, 11th International Conference on Concurrency Theory* 69–88.

Jensen, O. H. (2004) Forthcoming Ph.D. Thesis.

Jensen, O. H. and Milner, R. (2003) Bigraphs and transitions. In: *30th SIGPLAN-SIGACT Symposium on Principles of Programming Languages.*

Jensen, O. H. and Milner, R. (2004) Bigraphs and mobile processes (revised). Technical Report UCAM-CL-TR-580, University of Cambridge Computer Laboratory. (Available from `http://www.cl.cam.ac.uk/users/rm135`.)

Lawvere, F. W. (1963) Functorial semantics of algebraic theories. *Proc. Nat. Acad. Sci.* **50** 869–872.

Leifer, J. J. (2001) *Operational congruences for reactive systems*, Ph.D. Dissertation, University of Cambridge Computer Laboratory. Distributed in revised form as Technical Report 521. (Available from `http://pauillac.inria.fr/~leifer`.)

Leifer, J. J. and Milner, R. (2000) Deriving bisimulation congruences for reactive systems. *Proc. CONCUR 2000, 11th International Conference on Concurrency theory* 243–258. (Available at `http://pauillac.inria.fr/~leifer`.)

Leifer, J. J. and Milner, R. (2004) Transition systems, link graphs and Petri nets. Technical Report 598, University of Cambridge Computer Laboratory. (Available from `http://www.cl.cam.ac.uk/users/rm135`.)

Meseguer, J. and Montanari, U. (1990) Petri nets are monoids. *Information and Computation* **88** 105–155.

Milner, R. (1989) *Communication and Concurrency*, Prentice Hall.

Milner, R. (1996) Calculi for interaction. *Acta Informatica* **33** 707–737.

Milner, R. (2001a) Bigraphical reactive systems: basic theory. Technical Report 503, University of Cambridge Computer Laboratory. (Available from `http://www.cl.cam.ac.uk/users/rm135`.)

Milner, R. (2001b) Bigraphical reactive systems. Proc. 12th International Conference on Concurrency Theory. *Springer-Verlag Lecture Notes in Computer Science* **2154** 16–35.

Milner, R. (2004) Bigraphs for Petri nets. Advanced Course in Petri Nets. *Springer-Verlag Lecture Notes in Computer Science* **3098**.

Milner, R., Parrow, J. and Walker D. (1992) A calculus of mobile processes, Parts I and II. *Information and Computation* **100** 1–40 and 41–77.

Regev, A., Panina, E., Silverman, W., Cardelli, L. and Shapiro, E. (2004) BioAmbients: an abstraction for biological compartments. *Theoretical Computer Science* (to appear).

Priami, C., Regev, A., Silverman, W. and Shapiro, E. (2001) Application of a stochastic name-passing calculus to representation and simulation of molecular processes. *Information Processing Letters* **80** 25–31.

Sewell, P. (1998) From rewrite rules to bisimulation congruences. 9th International Conference on Concurrency Theory. *Springer-Verlag Lecture Notes in Computer Science* **1466** 269–284. (Full version to appear in *Theoretical Computer Science* **272**.)